



Research Paper

A generalized risk budgeting approach to portfolio construction

Martin Haugh, Garud Iyengar and Irene Song

Department of Industrial Engineering and Operations Research, Columbia University,
S. W. Mudd Building, 500 West 120th Street, New York, NY 10027, USA;
emails: h2078@columbia.edu, garud@ieor.columbia.edu, is2306@columbia.edu

(Received December 28, 2014; revised September 14, 2015; accepted December 16, 2015)

ABSTRACT

Risk-based asset allocation models have received considerable attention in recent years. This increased popularity is due in part to the difficulty in estimating expected returns, as well as to the 2008 financial crisis, which helped reinforce the key role of risk in asset allocation. We propose a generalized risk budgeting (GRB) approach to portfolio construction. In a GRB portfolio, assets are grouped into possibly overlapping subsets, and each subset is allocated a prespecified risk budget. Minimum variance, risk parity and risk budgeting portfolios are all special instances of a GRB portfolio. The GRB portfolio optimization problem is to find a GRB portfolio with an optimal risk–return profile, where risk is measured using any positively homogeneous risk measure. When the subsets form a partition, the assets all have the same expected return, and we restrict ourselves to long-only portfolios; then, the GRB problem can in fact be solved as a convex optimization problem. In general, however, the GRB problem is a constrained nonconvex problem, for which we propose two solution approaches. The first approach uses a semidefinite programming relaxation to obtain an (upper) bound on the optimal objective function value. In the second approach, we develop a numerical algorithm that integrates augmented Lagrangian and Markov chain Monte Carlo methods in order to find a point in the vicinity of a very good local optimum. This point is then supplied to a standard nonlinear optimization routine

Corresponding author: I. Song

Print ISSN 1460-1559 | Online ISSN 1755-2850
© 2017 Infopro Digital Risk (IP) Limited

with the goal of finding this local optimum. The merit of this second approach is in its generic nature: in particular, it provides a strategy for choosing a starting point for any nonlinear optimization algorithm.

Keywords: risk parity; risk budgeting; Markov chain Monte Carlo (MCMC); augmented Lagrangian (AL); portfolio optimization; semidefinite programming.

1 INTRODUCTION

Risk-based asset allocation models have received considerable attention in recent years. Some of this attention has been motivated by the difficulty in estimating expected returns. Mean–variance optimization, for example, is very sensitive to expected asset returns and, if applied naively, generally results in portfolios with extreme portfolio weights that are unstable over time. While there are now many methods for addressing these problems (see, for example, Black and Litterman 1992), there has been a trend of late to focus on approaches that are more robust to any assumptions on expected returns. The “ $1/N$ ” approach of DeMiguel *et al* (2009) is notable in this regard, as are the recent developments in risk-based asset allocation models, which are the focus of this paper.

As the term “risk-based” suggests, risk generally plays an important role in risk-based portfolio construction models. Examples of these models include the classic minimum variance approach of Markowitz and the more contemporary risk parity and risk budgeting approaches. In this study, we propose a generalized risk budgeting (GRB) approach to portfolio construction.

The concept of risk parity goes back to 1996, when Bridgewater Associates launched a risk parity fund called the All Weather Fund, although the term “risk parity” was only later coined by Qian (2006), who formalized the definition in terms of a risk budget, where weights of assets are determined in such a way that they all contribute equally to the overall portfolio risk. Maillard *et al* (2010) referred to such a portfolio as an equal risk contribution (ERC) portfolio. They analyzed properties of an unconstrained long-only ERC portfolio and showed that its volatility lies between the volatilities of the long-only minimum variance and equally weighted portfolios. We note here that the terms “risk parity” and “equal risk contribution” are used interchangeably in the literature, but hereafter we will use the former.

A risk parity portfolio is not always desirable, however. An investor may prefer to allocate different risk budgets to each asset, and this preference would require a more general risk budgeting portfolio. Theoretical properties of risk budgeting portfolios were analyzed by Bruder and Roncalli (2012). Extending the result of Maillard *et al* (2010), they showed that the volatility of a long-only risk budgeting portfolio lies between the volatilities of a long-only minimum variance portfolio and a long-only

weighted portfolio whose weights are proportional to their risk budgets. They further demonstrated that, when the portfolio risk is computed using a convex risk measure and risk budgets are defined to be strictly positive, a long-only risk budgeting portfolio exists and is unique.

Since the introduction of this approach, there have been many additional studies on risk parity and risk budgeting approaches. Most of them, however, have focused on seeking a long-only minimal risk portfolio that satisfies (predefined) risk budgeting constraints. Most of these methods, therefore, lack flexibility. For example, by disregarding the expected asset returns in their problem formulations, many of these methods make the implicit assumption that all asset returns are identical in expectation. Regardless of whether disregarding expected returns results in a better-performing portfolio, such an assumption does not hold in practice. In addition, it is clearly desirable that investors be able to freely express their views on expected asset returns when constructing a portfolio.¹

In this paper, we propose a GRB problem formulation that leads in general to a nonconvex optimization problem. We refer to this problem as the GRB portfolio optimization problem (or the GRB problem). We then develop solution approaches for this GRB problem. The key advantage of our formulation over the prevailing risk parity or risk budgeting approaches is that it offers a much greater degree of flexibility in the way risk-based portfolios are constructed. It allows short sales of assets and the use of risk factors to model asset returns and, most importantly, it allows investors to define risk budgets for overlapping subsets of assets.

When the subsets form a partition, the assets all have the same expected return and we restrict ourselves to long-only portfolios, we show that the problem can be formulated as a convex optimization problem and, therefore, easily solved. This result generalizes the approach of Bruder and Roncalli (2012) for constructing a long-only risk budgeting portfolio with minimum variance. For the more general GRB problem, we propose two solution approaches. The first is a semidefinite programming (SDP) relaxation of the problem, which allows us to obtain an (upper) bound on the optimal objective function of the GRB problem. We remark that the solution to this SDP problem is also often a very good starting point for a generic nonlinear optimization solver. To the best of our knowledge, there are no other studies that apply an SDP

¹ Research that addresses the issue of incorporating expected asset returns into risk constrained portfolio selection includes Zhu *et al* (2010, 2011) and Boudt *et al* (2012). Zhu *et al* (2010) studied optimal mean–variance marginal risk constrained portfolio selection with the purpose of controlling relative risk contribution of individual assets. Zhu *et al* (2011) presented a factor-risk-constrained mean–variance portfolio optimization model, and Boudt *et al* (2012) proposed a minimum conditional value-at-risk (CVaR) concentration portfolio that balances the investor’s return objective and the diversification of risk across portfolio constituents.

relaxation to the risk parity or risk budgeting problems. Our second approach develops a numerical algorithm that combines augmented Lagrangian (AL) and Markov chain Monte Carlo (MCMC) methods, with the goal of finding a point in the vicinity of a very good local optimum. This point is then supplied to a nonlinear optimization routine to compute this local optimum. The merit of this second approach is in its generic nature: in particular, it provides a strategy for choosing a starting point for any nonlinear optimization algorithm.

The remainder of the paper is organized as follows. In Section 2, we formally define the GRB problem and describe our two solution approaches as well as a special case that can be solved as a convex optimization problem. We provide numerical and empirical results for the SDP relaxation and the AL–MCMC approach in Section 3, and discuss some of the practical challenges associated with applying the latter approach in Section 4. We then conclude in Section 5.

2 THE GENERALIZED RISK BUDGETING PROBLEM

In portfolio construction and analysis, it is often preferable to group assets according to their attributes, such as asset class, country, sector and industry. In the case of an investment portfolio with a broad coverage of asset classes, for example, it may be more insightful and therefore preferable to look at the marginal risk contribution of each asset class rather than each individual asset in the portfolio. Similarly, in the case of a large-scale stand-alone asset class portfolio, it may be preferable to control the risk contributions of assets at an aggregate level such as country or market sector. The GRB strategy is based on this very idea of managing the marginal risk contributions of subsets of assets to the total portfolio risk. In a GRB portfolio, the risk contribution from each (prespecified) subset of assets is set equal to some prespecified risk budget. Note that we are using the term “subset” rather than “partition” since, depending on the attributes used for the asset classification, assets may belong to more than one group. We will see later that minimum variance, risk parity and risk budgeting portfolios are all special instances of a GRB portfolio.

The objective of the GRB problem is to find a GRB portfolio that is optimal on the basis of its risk–return profile. Portfolio risk in the GRB problem is computed via a positively homogeneous risk measure, for which we can use Euler’s theorem to provide a risk decomposition. Examples of positively homogeneous risk measures include portfolio volatility, value-at-risk (VaR) and any coherent risk measures such as CVaR (Artzner *et al* 1999).

Toward this end, let $\mathcal{R}(x) : \mathbb{R}^d \rightarrow \mathbb{R}$ denote a generic risk measure that is a positively homogeneous function of degree 1 in the portfolio weight vector, x . Euler’s

theorem then provides the following additive risk decomposition:

$$\mathcal{R}(x) = \sum_{i=1}^d x_i \frac{\partial \mathcal{R}(x)}{\partial x_i}, \quad (2.1)$$

where the marginal risk contribution of the i th asset is

$$\text{RC}_i(x) = x_i \frac{\partial \mathcal{R}(x)}{\partial x_i}.$$

If $\mathcal{M}_1, \dots, \mathcal{M}_s \subseteq \{1, \dots, d\}$ denote s subsets of portfolio assets,² then the marginal risk contribution of the k th subset is

$$\text{RC}_{\mathcal{M}_k}(x) := \sum_{i \in \mathcal{M}_k} \text{RC}_i(x).$$

Let β_1, \dots, β_s now denote the risk budgets for $\mathcal{M}_1, \dots, \mathcal{M}_s$, respectively. We can then formulate the GRB problem as

$$\max_{x \in \mathcal{X}} \mu'x - \lambda \mathcal{R}(x), \quad \text{subject to } \sum_{i \in \mathcal{M}_k} \text{RC}_i(x) = \beta_k \mathcal{R}(x), \quad k = 1, \dots, s, \quad (\text{GRB})$$

where $\mu \in \mathbb{R}^d$ is a vector of expected returns, λ is a risk aversion parameter and $\mathcal{X} := \{x \in \mathbb{R}^d : \mathbf{1}'x = 1\}$. Note that the constraint $\sum_{i \in \mathcal{M}_k} \text{RC}_i(x) = \beta_k \mathcal{R}(x)$ implies that $\sum_{k=1}^s \beta_k = 1$ when the \mathcal{M}_k form a partition. We note that the GRB problem becomes a minimum variance problem when $\mu = \mu_0 \mathbf{1}$; there is only one subset, \mathcal{M}_1 , which is equal to the universe of assets, and the risk measure is portfolio volatility. It is a risk parity problem when $\mu = \mu_0 \mathbf{1}$, the \mathcal{M}_k are all singletons and all risk budgets, β_k , are equal. Finally, it is a risk budgeting problem when $\mu = \mu_0 \mathbf{1}$ and the \mathcal{M}_k are again all singletons.

The GRB problem is a constrained nonconvex optimization problem for which efficient solution algorithms are unavailable. Although there are numerous methods available for computing risk parity portfolio weights (see, for example, Bai *et al* 2016; Spinu 2013), these methods are in general not applicable to the GRB problem. In Section 2.1, we consider a special case of the GRB problem that can be solved as a convex optimization problem. We then proceed to discuss our solution approaches for the general nonconvex case. Note that the parameter and variable notation introduced in this section will be used throughout the paper unless otherwise stated.

² In practice, the subsets \mathcal{M}_k would typically correspond to the different asset classes included in an investment portfolio. For example, AQR Risk Parity funds contain approximately ninety assets from six different asset classes: fixed income, treasury inflation protected securities, equity, currency, commodity and credit. The size of each asset class in these funds ranges from five to twenty-five securities. For the complete holdings of AQR Risk Parity funds, we refer the reader to <http://bit.ly/2iAGliK>.

2.1 A special case of the GRB problem

We now consider a special case of the GRB problem in which all assets have the same expected return, ie, $\mu = \mu_0 \mathbf{1}$, each asset belongs to one and only one subset, ie, the \mathcal{M}_k form a partition of the asset space, and nonnegativity constraints are imposed at the partition level. We can then reformulate the GRB problem as follows:

$$\min_{x \in \mathcal{X}} \mathcal{R}(x), \quad \text{subject to} \quad \sum_{i \in \mathcal{M}_k} \text{RC}_i(x) = \beta_k \mathcal{R}(x), \quad \sum_{i \in \mathcal{M}_k} x_i \geq 0, \quad k = 1, \dots, s. \quad (2.2)$$

Assuming also that each $\beta_k > 0$, we then have the following result that extends Bruder and Roncalli (2012).

THEOREM 2.1 *Assuming $\mathcal{R}(y) \neq 0$ for nonzero y , (2.2) is equivalent to the convex optimization problem:*

$$\min_y \mathcal{R}(y), \quad \text{subject to} \quad \sum_{k=1}^s \beta_k \ln \left(\sum_{i \in \mathcal{M}_k} y_i \right) \geq c, \quad (2.3)$$

where c is an arbitrary constant. In particular, the normalized optimal solution \tilde{y}^* to (2.3) is also the optimal solution to (2.2). (A normalized solution is one where $\sum_{i=1}^d \tilde{y}_i^* = 1$; see the discussion after the proof.)

PROOF Let $\mathcal{L}(y, \gamma)$ denote the Lagrangian of the optimization problem (2.3), so that

$$\mathcal{L}(y, \gamma) = \mathcal{R}(y) - \gamma \left(\sum_{k=1}^s \beta_k \ln \left(\sum_{i \in \mathcal{M}_k} y_i \right) - c \right).$$

At optimality, the solution y^* satisfies the Karush–Kuhn–Tucker conditions. That is, y^* satisfies

- (i) the first-order conditions

$$\frac{\partial \mathcal{L}(y, \gamma)}{\partial y_i} = \frac{\partial \mathcal{R}(y)}{\partial y_i} - \gamma \left(\frac{\beta_k}{\sum_{j \in \mathcal{M}_k} y_j} \right) = 0 \quad (2.4)$$

for $i = 1, \dots, d$, where k is the index of the subset \mathcal{M}_k containing i ;

- (ii) the complementary slackness conditions

$$\gamma \left(\sum_{k=1}^s \beta_k \ln \left(\sum_{i \in \mathcal{M}_k} y_i \right) - c \right) = 0. \quad (2.5)$$

Note that, as $\ln: \mathbb{R}^+ \rightarrow \mathbb{R}$, we must have $\sum_{i \in \mathcal{M}_k} y_i > 0$ for $k = 1, \dots, s$, and hence y cannot be $\mathbf{0}$. Then, since $\mathcal{R}(y) \neq 0$ for nonzero y , at least one of $\partial \mathcal{R}(y)/\partial y_i$ must be nonzero by (2.1). The strict positivity of the β_k and (2.4) then imply $\gamma > 0$. We therefore have

$$\frac{\partial \mathcal{R}(y)}{\partial y_i} = \gamma \left(\frac{\beta_k}{\sum_{j \in \mathcal{M}_k} y_j} \right) \quad (2.6)$$

for $i = 1, \dots, d$, and where $\gamma > 0$. Multiplying both sides of (2.6) by y_i and then summing over $i \in \mathcal{M}_k$ yields

$$\begin{aligned} \sum_{i \in \mathcal{M}_k} y_i \frac{\partial \mathcal{R}(y)}{\partial y_i} &= \gamma \left(\frac{\beta_k}{\sum_{j \in \mathcal{M}_k} y_j} \right) \sum_{i \in \mathcal{M}_k} y_i \\ &= \gamma \beta_k \end{aligned}$$

for $k = 1, \dots, s$. We therefore see that the risk contribution of each \mathcal{M}_k is proportional to its risk budget, β_k . The normalized optimal solution \tilde{y}^* is then the optimal solution x^* to (2.2), as claimed. \square

Note that as

$$\sum_{k=1}^s \beta_k \ln \left(\sum_{i \in \mathcal{M}_k} y_i \right) = c$$

by (2.5), we could directly obtain x^* from (2.3) if we used $c^* = c - \ln(\sum_{i=1}^d y_i)$, rather than the original c that led to the solution y . Note also that we recover the results of Bruder and Roncalli (2012) if the \mathcal{M}_k are all singletons.

2.2 An SDP relaxation for the general GRB problem

Our first approach to the GRB problem uses a semidefinite programming relaxation to obtain an upper bound on the optimal objective function value. There are two advantages of the SDP approach:

- (i) the solution to the SDP problem (which is generally infeasible for the GRB problem) can be used as a (hopefully very good) starting point for a standard nonlinear optimization routine;
- (ii) the SDP solution can often provide a ‘‘certificate’’ of near-optimality when the SDP solution has an objective function that is close to the objective function of the best local optimal solution that we have found.

In our development of the SDP approach, we will assume initially that our risk measure is portfolio volatility, so that

$$\mathcal{R}(x) := \sqrt{x' \Sigma x},$$

where $\Sigma \in \mathbb{R}^{d \times d}$ is the covariance matrix of asset returns. The marginal risk contributions of the individual assets then satisfy

$$\text{RC}_i(x) = x_i \frac{(\Sigma x)_i}{\sqrt{x' \Sigma x}}, \quad i = 1, \dots, d.$$

With this measure of risk, we can rewrite the GRB problem in the following equivalent form:

$$\begin{aligned} \max_{x, X} \quad & \mu'x - \lambda \mathcal{R}(x), \\ \text{subject to} \quad & \sum_{i \in \mathcal{M}_k} \text{tr}(\Gamma_i X) = \beta_k \text{tr}(\Sigma X), \quad k = 1, \dots, s, \quad X = xx', \quad \mathbf{1}'x = 1, \end{aligned} \quad (2.7)$$

where $\Gamma_i = e_i e_i' \Sigma$. Here, e_i denotes the i th column of the identity matrix $I \in \mathbb{R}^{d \times d}$, $x \in \mathbb{R}^d$, and $\text{tr}(\cdot)$ denotes the trace of a matrix. Since $X = xx'$ is the only nonconvex constraint in (2.7), we obtain a convex relaxation of the GRB problem by relaxing this constraint to $X \succeq xx'$. We then obtain the following SDP relaxation of our GRB problem:

$$\max_{x, X} \mu'x - \lambda \mathcal{R}(x), \quad (2.8)$$

subject to

$$\sum_{i \in \mathcal{M}_k} \text{tr}(\Gamma_i X) = \beta_k \text{tr}(\Sigma X), \quad k = 1, \dots, s, \quad \begin{bmatrix} X & x \\ x' & 1 \end{bmatrix} \succeq 0, \quad \mathbf{1}'x = 1,$$

where we have used the Schur complement to reformulate the semidefinite constraint $X \succeq xx'$ as a linear matrix inequality. Note that we can recover (2.7) from (2.8) by imposing an additional (nonconvex) constraint that the left-hand-side of the linear matrix inequality in (2.8) be a rank-1 matrix. The SDP relaxation can be solved efficiently and the SDP solution provides an upper bound on the optimal objective function of the GRB problem. For example, we can easily implement and solve (2.8) using CVX (Grant and Boyd 2008, 2014).³

SDP relaxations can also be formulated for the GRB problem with other risk measures. Suppose the risk measure is either the VaR or CVaR of a portfolio. Let $F_r(z) := P\{r \leq z\}$ denote the cumulative distribution function of the portfolio return r . Then, the VaR and CVaR at the confidence level $\alpha \in (0, 1)$ are defined as

$$\text{VaR}_\alpha := \min\{z \mid F_r(z) \geq \alpha\}$$

³ A special SDP mode in CVX allows positive (negative) semidefinite constraints \succeq (\preceq) to be imposed using MATLAB's standard inequality operators $>=$ ($<=$).

and

$$\text{CVaR}_\alpha := E[r \mid r \geq \text{VaR}_\alpha(r)].$$

Suppose the asset returns are, as before, normally distributed, with mean vector μ and covariance matrix Σ . Then, for a portfolio with the weight vector x , it easily follows that

$$\text{VaR}_\alpha(x) = \mu'x + \Psi_1(\alpha)\sqrt{x'\Sigma x}$$

and

$$\text{CVaR}_\alpha(x) = \mu'x + \Psi_2(\alpha)\sqrt{x'\Sigma x},$$

where (see McNeil *et al* 2005)

$$\begin{aligned}\Psi_1(\alpha) &= \sqrt{2} \operatorname{erf}^{-1}(2\alpha - 1), \\ \Psi_2(\alpha) &= (\sqrt{2\pi} \exp(\operatorname{erf}^{-1}(2\alpha - 1))^2(1 - \alpha))^{-1}, \\ \operatorname{erf}(z) &= \left(\frac{2}{\sqrt{\pi}}\right) \int_0^z e^{-t^2} dt.\end{aligned}$$

Likewise, the respective marginal VaR and CVaR contributions of the i th asset are given by⁴

$$\text{RC}_{i,\text{VaR}_\alpha}(x) = \mu_i + x_i \frac{(\Sigma x)_i}{\sqrt{x'\Sigma x}} \Psi_1(\alpha)$$

and

$$\text{RC}_{i,\text{CVaR}_\alpha}(x) = \mu_i + x_i \frac{(\Sigma x)_i}{\sqrt{x'\Sigma x}} \Psi_2(\alpha).$$

Without loss of generality, let us consider the risk budgeting constraints that arise when we use VaR as our risk measure. These take the form

$$\sum_{i \in \mathcal{M}_k} \left(\mu_i + x_i \frac{(\Sigma x)_i}{\sqrt{x'\Sigma x}} \Psi_1(\alpha) \right) = \beta_k (\mu'x + \Psi_1(\alpha)\sqrt{x'\Sigma x}), \quad k = 1, \dots, s. \quad (2.9)$$

The VaR version of the GRB problem is given by

$$\max_{x, X} \mu'x - \lambda \text{VaR}_\alpha(x), \quad (2.10)$$

subject to

$$\sum_{i \in \mathcal{M}_k} (\mu_i \sqrt{x'\Sigma x} + x_i (\Sigma x)_i \Psi_1(\alpha)) = \beta_k (\mu'x \sqrt{x'\Sigma x} + \Psi_1(\alpha) x' \Sigma x),$$

$$k = 1, \dots, s,$$

$$X = xx', \quad \mathbf{1}'x = 1.$$

⁴ See Boudt *et al* (2012), who also studied portfolio selection under CVaR budgets.

Note that the risk budgeting constraints of (2.10) are obtained by multiplying both sides of (2.9) by $\sqrt{x' \Sigma x}$. Let $t = \sqrt{x' \Sigma x}$. Then, the optimization problem (2.10) can be reformulated as follows:

$$\max_{t, \delta, w, x, y, X, Z} \mu' x - \lambda \text{VaR}_\alpha(x), \quad (2.11)$$

subject to

$$\sum_{i \in \mathcal{M}_k} (\mu_i t + \text{tr}(\Gamma_i X) \Psi_1(\alpha)) = \beta_k (\mu' \delta + \Psi_1(\alpha) \text{tr}(\Sigma X)), \quad k = 1, \dots, s,$$

$$Z = \begin{bmatrix} 1 & t & x' \\ t & y & \delta' \\ x & \delta & X \end{bmatrix}, \quad \text{rank}(Z) = 1,$$

$$t = \sqrt{x' \Sigma x}, \quad t, y \geq 0, \quad x, \delta \in \mathbb{R}^d, \quad X \in \mathbb{R}^{d \times d}, \quad Z \succeq 0,$$

where we use the fact that $\text{rank}(Z) = 1$ implies that $\delta = tx$, and $X = xx'$. There are two nonconvex constraints in (2.11): $\text{rank}(Z) = 1$ and $t = \sqrt{x' \Sigma x}$. We can obtain a semidefinite relaxation for the feasible set by dropping the $\text{rank}(Z) = 1$ constraint and relaxing the other constraint to $t \geq \sqrt{x' \Sigma x}$.⁵ Since $\text{VaR}_\alpha(x)$ is not a convex function of x , we will have to replace $\text{VaR}_\alpha(x)$ by a concave function that is pointwise larger in order to obtain a valid relaxation for (2.11). This last step would not be necessary if we used CVaR as our risk measure because $\text{CVaR}(x)$ is already a convex function of x . We also note that, while we have assumed normally distributed asset returns here, other distributions such as the t -distribution could be used instead. See Boyd and Vandenberghe (1997) for SDP relaxations of nonconvex problems more generally.

2.3 An AL–MCMC approach

Our second approach to solving the GRB problem involves combining the AL approach with MCMC sampling to generate a point in the proximity of the global optimum of the GRB problem. This can then be used as a starting point for a nonlinear optimization routine to converge to a globally optimal GRB portfolio. The underlying idea of the algorithm is to effectively sample points with a higher objective function value and simultaneously drive the sample path in the direction of the feasible region using the AL terms.

Let Ω be the state space and let $p(x) = C^{-1} p^*(x)$ denote some target probability distribution on Ω , where $C := \int_{\Omega} p^*(x) dx$ is the normalization constant. The MCMC method is an approach used to sample from $p(x)$ when the normalizing

⁵This is a second-order cone constraint that can in turn be reformulated as a semidefinite constraint.

constant is hard to compute. In this approach, we construct a Markov chain on Ω using a “proposal” distribution $q(x_{t+1} | x_t)$ in such a way that $p(x)$ is the unique stationary distribution for the Markov chain. Modulo some technical conditions, the main requirement of MCMC is that the unnormalized distribution, $p^*(x)$, should be easy to compute.⁶ Given a current sample x_t at time t , the proposal distribution, $q(\cdot | x_t)$, is used to generate a candidate sample, x_{t+1} , which is then accepted with probability

$$\alpha(x_t, x_{t+1}) := \min \left\{ 1, \frac{q(x_t | x_{t+1}) p^*(x_{t+1})}{q(x_{t+1} | x_t) p^*(x_t)} \right\}. \quad (2.12)$$

If the candidate point x_{t+1} is rejected, we then set $x_{t+1} = x_t$ and continue sampling in this manner.

Since our goal is to solve the GRB problem, one possibility would be to set

$$p^*(x) = \exp(\gamma F(x)) \mathbb{I}_{\mathcal{F}}(x),$$

where $F(x) := \mu'x - \lambda \mathcal{R}(x)$ denotes the objective function of the GRB problem, γ is an annealing parameter that is used to concentrate the $p^*(x)$ in the proximity of the global optimum and $\mathbb{I}_{\mathcal{F}}(\cdot)$ denotes the indicator function of the set

$$\mathcal{F} = \left\{ x \in \mathcal{X} \mid h_k(x) := \sum_{i \in \mathcal{M}_k} \text{RC}_i(x) - \beta_k \mathcal{R}(x) = 0, k = 1, \dots, s \right\}.$$

Since the feasible region \mathcal{F} of the GRB problem is, typically, very “small”, $p^*(x_{t+1})$ is likely to be zero for most candidate points x_{t+1} , and these points will be rejected in the acceptance–rejection step (2.12). Therefore, using MCMC to sample only from the feasible region is very difficult, particularly for high-dimensional problems.

One possible approach to overcoming these difficulties is to allow the MCMC iterates x_t to be infeasible, but to “direct” them toward the feasible region by adding a term that penalizes infeasibility to our definition of p^* . In particular, we could define

$$P_c(x) := F(x) + \frac{1}{2}c \|h(x)\|_2^2,$$

where c is a negative constant, and now use $p^*(x) = e^{\gamma P_c(x)}$ as the unnormalized density. The main difficulty with the penalty approach is that it is very sensitive to the value of the penalty parameter, c . This is a well-known phenomenon, and the augmented Lagrangian algorithm was introduced in order to circumvent this numerical instability.

⁶ See Robert and Casella (2004) for further technical details on MCMC algorithms.

The AL-MCMC algorithm

In the augmented Lagrangian approach, we define the time- t target distribution⁷ to be $p_t^*(x) := \exp(\gamma_t \mathcal{L}_{c_t}(u_t, x))$, where the augmented Lagrangian function of the GRB problem is defined as

$$\begin{aligned} \mathcal{L}_{c_t}(u_t, x) &:= F(x) + u_t' h(x) + \frac{1}{2} c_t \|h(x)\|_2^2 \\ &= \mu' x - \lambda \mathcal{R}(x) + \sum_{k=1}^s u_{t,k} \left(\beta_k \mathcal{R}(x) - \sum_{i \in \mathcal{M}_k} \text{RC}_i(x) \right) \\ &\quad + \frac{c_t}{2} \left(\sum_{k=1}^s \left(\beta_k \mathcal{R}(x) - \sum_{i \in \mathcal{M}_k} \text{RC}_i(x) \right)^2 \right), \end{aligned}$$

where $u_t = (u_{t,1}, \dots, u_{t,s}) \in \mathbb{R}^s$ is a vector of time- t Lagrange multipliers.⁸ Let $d_{c_t}(u) := \max_{x \in \mathcal{X}} \mathcal{L}_{c_t}(u, x)$ denote the dual objective.

The initial vector of Lagrange dual multipliers, u_0 , and the penalty parameter, c_0 , are specified exogenously. The values for dual multipliers, u_t , and the nonincreasing penalty parameter, c_t , for $t \geq 1$ are chosen adaptively during the course of the simulation. In particular, we decrease c_t by a predetermined value ε_c when there is no improvement in constraint violations over a particular iteration. When there is an improvement in constraint violation, we do not update c_t , but instead update the Lagrange multipliers using the first-order conditions, ie, we set

$$u_{t+1} = u_t - \varepsilon_u \nabla d_{c_t}(u_t), \quad (2.13)$$

where $d_{c_t}(u)$ denotes the dual function, and ε_u is a given step size. We chose not to update both c_t and u_t in every iteration in order to ensure that we leave the current location only after adequately exploring its neighborhood. The update of duals u_t or the penalty parameter c_t occurs at every iteration, whether the candidate x_{t+1} is accepted or not. We note that (2.13) represents the steepest descent iteration for minimizing d_{c_t} , but we may choose other methods, such as Newton's method, for updating the Lagrange multipliers (see Appendix A online). Furthermore, we can use other criteria for updating c_t . See Bertsekas (1996) for a detailed discussion of the augmented Lagrangian method.

To generate a candidate value of x_{t+1} , we use a proposal distribution based on a random walk chain. In particular, we generate $z_t^* \sim N(0, \sigma_t^q I)$ and take

$$x_{t+1} = x_t + z_t^*$$

⁷ We note that, since p_t^* now changes with each iteration, there is no longer a fixed target stationary distribution for our algorithm.

⁸ See Appendix A for further details on the augmented Lagrangian functions.

as our candidate point, which is then accepted with probability $\alpha(x_t, x_{t+1})$. The standard deviation of the proposal distribution σ_t^q can be thought of as a tuning parameter that we can adjust to increase the acceptance probability when the current location is near a feasible region. This is done by decreasing σ_t^q by a factor of κ , where $0 < \kappa < 1$, only if the percentage drop in the size of the constraint violations is larger than a predetermined value, δ .⁹ The value of κ can thus be interpreted as the rate at which we decrease σ_t^q when the current location is considered favorable. If κ is too small, then σ_t^q decreases too fast and moves become small too quickly. Small moves are generally accepted (high acceptance probability), and as a result the chain is prone to get trapped in the current region prior to exploring other regions. On the contrary, if κ is too big, then σ_t^q decreases too slowly and moves remain large. This leads to many rejections, resulting in an inefficient chain.¹⁰

In each iteration, irrespective of whether the proposed sample x_{t+1} is accepted or rejected, the annealing parameter γ_t is increased according to

$$\gamma_t = \sigma_\gamma \gamma_{t-1}, \quad (2.14)$$

where $\sigma_\gamma = (\gamma_{\max}/\gamma_0)^{1/T}$ and γ_{\max} is the maximum permitted value of γ_t . Note that (2.14) is known as the geometric annealing schedule.¹¹ Thus, the AL-MCMC algorithm is a simulated annealing algorithm (see Van Laarhoven and Arts (1987) for further details) where, by forcing $\lim_{t \rightarrow \infty} \gamma_t = \infty$, we hope to drive samples toward the global optimum of the GRB problem.

The AL-MCMC algorithm attempts to combine the best aspects of the augmented Lagrangian method and the MCMC method. The augmented Lagrangian term guides the Markov chain toward a feasible region, while the acceptance–rejection step in the MCMC method attempts to ensure that the iterates do not get trapped in poor local maxima of the GRB problem.

⁹ See Algorithm 2.2 for precise details. Depending on the specific problem under consideration, we may choose to modify this step or simply to keep σ_t^q constant across all t . However, considering that choosing the acceptance rate is important for a good numerical performance of the algorithm, it is recommended that σ_t^q is allowed to vary with t . Also, instead of having the annealing schedule, we can directly adjust σ_t^q based on the acceptance rate at time t . This is in fact considered common practice for adjusting the mixing and the acceptance probability of the chain. The basic idea is to increase σ_t^q when the acceptance rate is too high and decrease σ_t^q when the acceptance rate is too low. We refer the reader to Roberts *et al* (1994) for further details.

¹⁰ In choosing κ , we generated a training data set for each test case presented in Section 3, and tested multiples of 0.25 as candidate values of κ . Based on this, we selected $\kappa = 0.75$ across all test cases.

¹¹ The geometric annealing schedule, the most commonly used annealing schedule in practice, was also suggested by the originators of the simulated annealing algorithm (Kirkpatrick *et al* 1983). Depending on the specific problem under consideration, we may choose to use a different annealing schedule. For a comparison of different annealing schedules, we refer the reader to Nourani and Andresen (1998).

A complete specification of our AL-MCMC algorithm is given in Algorithm 2.2. A feasible sample point with the highest value of $F(\cdot)$ is probably most suitable to be used as a starting point for a nonlinear optimization routine. However, as the direct sampling of a feasible point is overly difficult for the GRB problem, the last point obtained by the algorithm is then fed to a nonlinear optimization routine with the goal of quickly finding a good nearby local maximum.

We note that our algorithm is a heuristic algorithm that we hope is capable of producing good starting points for a nonlinear optimization solver. We expect this algorithm to be useful for general nonconvex optimization problems beyond the GRB problem of this paper. There is also further scope for improvement. For example, we could use a more sophisticated MCMC algorithm compared with Metropolis–Hastings. For example, if we suspect that $F(\cdot)$ or $L_{c_t}(\cdot)$ is multimodal, then hybrid MCMC methods such as the Hamiltonian MCMC should be superior. It is also possible to tailor the proposal distributions, $q(x_{t+1} | x_t)$, for the problem at hand. Note also that, while not stated explicitly, it of course makes sense to keep track of the best feasible sample that has been obtained during the execution of the algorithm.

We therefore propose the following procedure to solve the GRB problem.

STEP 1 Generate an initial vector x_0 to be used as the starting point of the Markov chain and choose values of $\gamma_0, \sigma_\gamma, \varepsilon_c, \varepsilon_u, \delta, \sigma_0^q, c_0, u_0$ and κ to be used as parameters for the AL-MCMC algorithm (Algorithm 2.2).

STEP 2 Perform the AL-MCMC algorithm to obtain an initial point x_s to be fed to a nonlinear optimization routine.

STEP 3 Solve the GRB problem using a nonlinear optimization solver with x_s obtained from step 1 as the initial guess.

The AL-MCMC algorithm described in this section can be further enhanced by using a set of different random starting points x_0 for generating Markov chains. For instance, in our numerical experiments we used antithetic starting points to generate several values of x_s .¹²

ALGORITHM 2.2 (AL-MCMC)

1. Choose $x_0, \gamma_0, \sigma_\gamma, \varepsilon_c, \varepsilon_u, \delta, \sigma_0^q, c_0, u_0, \kappa$
2. **for** $t = 0 : T$ **do**
3. Draw a candidate sample x_{t+1} from the proposal $q(x_{t+1} | x_t)$

¹² Readers interested in antithetic variates in Monte Carlo techniques can refer to Robert and Casella (2004).

4. Let $\alpha(x_t, x_{t+1}) = \min \left\{ 1, \frac{q(x_t | x_{t+1})p^*(x_{t+1})}{q(x_{t+1} | x_t)p^*(x_t)} \right\}$,
 where $\frac{p^*(x_{t+1})}{p^*(x_t)} = e^{\gamma_t(\mathcal{L}_{c_t}(u_t, x_{t+1}) - \mathcal{L}_{c_t}(u_t, x_t))}$.
5. **if** $\alpha \geq 1$ **then**
6. $x_{t+1} \leftarrow x_{t+1}$ # Accept the candidate
7. **else**
8. Draw $p \sim \mathbb{U}[0, 1]$
9. **if** $p \leq \alpha$ **then**
10. $x_{t+1} \leftarrow x_{t+1}$ # Accept the candidate
11. **else**
12. $x_{t+1} \leftarrow x_t$ # Reject the candidate
13. **end if**
14. **end if**
15. $\gamma_{t+1} \leftarrow \sigma_\gamma \gamma_t$ # Update the annealing parameter
16. **if** $\|h(x_{t+1})\|_2^2 < \|h(x_t)\|_2^2$ **then**
17. $u_{t+1} \leftarrow u_t - \varepsilon_u \nabla d_{c_t}(u_t)$, where $d_{c_t}(u) = \max_{x \in \mathcal{X}} \mathcal{L}_{c_t}(u, x)$
 # Update the Lagrange multipliers
18. **if** $\frac{\|h(x_t)\|_2^2}{\|h(x_t + 1)\|_2^2} - 1 > \delta$ **then**
19. $\sigma_{t+1}^q \leftarrow \kappa \sigma_t^q$ # Update the jump size
20. **end if**
21. **else**
22. $c_{t+1} \leftarrow c_t + \varepsilon_c$ # Update the penalty parameter
23. **end if**
24. **end for**

3 NUMERICAL RESULTS

We now present numerical results for the two proposed approaches: the SDP relaxation and the AL-MCMC algorithm. We first describe a simple example to illustrate the potential effectiveness of the AL-MCMC algorithm. We then discuss the performances of the two approaches when they are tested on GRB problems with the number of assets ranging from 7 to 200. All the results presented in this section are based on percentage returns, ie, returns are multiplied by 100, unless otherwise stated. Note also that the term “optimal solution” generally denotes a local optimum.

3.1 Numerical results for a small example

Our first problem¹³ is a five-asset problem with a variance–covariance matrix of percentage returns:

$$\Sigma = \begin{bmatrix} 94.868 & 33.750 & 12.325 & -1.178 & 8.778 \\ 33.750 & 445.642 & 98.955 & -7.901 & 84.954 \\ 12.325 & 98.955 & 117.265 & 0.503 & 45.184 \\ -1.178 & -7.901 & 0.503 & 5.460 & 1.057 \\ 8.778 & 84.954 & 45.184 & 1.057 & 34.126 \end{bmatrix}.$$

We also assumed that the expected returns of these assets are identical, so that $\mu = \mu_0 \mathbf{1}$. Suppose now we want to compute a long-only risk parity portfolio with minimum variance, and that we apply the AL-MCMC algorithm to solve this problem. We used a single Markov chain of 5000 points, ie, $T = 5000$ in Algorithm 2.2, and x_0 was generated uniformly from the five-dimensional unit cube. We also used the following parameters:

- initial annealing parameter $\gamma_0 = 1$ with $\sigma_\gamma = 1.0007$;
- initial penalty parameter $c_0 = -10\,000$ with $\varepsilon_c = 0$;
- jump size $\sigma_0^q = 0.5$ with $\kappa = 0.75$;
- threshold parameter for updating σ_t^q , $\delta = 0.01$;
- initial Lagrange multipliers $u_0 = \mathbf{0}$ with $\varepsilon_u = 0.01$.

Since this problem is relatively simple with just five constraints, we did not need to update c during the course of the algorithm. $x_{5000} = [0.1245; 0.0467; 0.0833; 0.6133; 0.1323]$ is the last point obtained from the AL-MCMC algorithm. If we specify the feasibility tolerance to 10^{-4} , this point is, in fact, the optimal risk

¹³ This is the example presented in Bai *et al* (2016).

parity solution.¹⁴ Without the use of a nonlinear optimization routine, the AL-MCMC algorithm was therefore able to discover a good risk parity solution. The running time for the algorithm was 1.31 seconds.¹⁵

When the algorithm was applied without the penalty parameter, ie, $c_t = 0$ for all t , or without the Lagrange multipliers, ie, $u_t = \mathbf{0}$ for all t , it failed to converge to a risk parity solution. All of its sampled points violated the risk parity constraints by more than 10^{-4} , and hence the help of a nonlinear optimization routine was necessary for finding an optimal risk parity solution. When its last point was supplied to a nonlinear optimization routine, the optimal risk parity solution was found successfully. These results demonstrate the potential advantage of incorporating the augmented Lagrangian method into the MCMC algorithm.

3.2 Numerical results for the GRB problem

For more general GRB problems we focused on the portfolio volatility risk measure $\mathcal{R}(x) := \sqrt{x' \Sigma x}$ and assumed a risk aversion parameter $\lambda = 1$. Expected asset returns μ , covariance matrixes Σ and risk budgets β were all generated randomly. In particular, we sampled μ from $N(0, I)$, and for Σ we first generated a matrix $V \in \mathbb{R}^{d \times d}$ using a standard normal distribution and then converted it into a symmetric positive semidefinite matrix by multiplying it by its transpose, ie, $\Sigma = V'V$. We generated risk budgets $\beta = (\beta_1, \dots, \beta_s)$ from $\mathbb{U}^s(0, 1)$ and normalized them such that $\sum_{k=1}^s \beta_k = 1$.

We considered the five test cases listed in Table 1 under two different scenarios. In the first scenario, we assumed that $\mu = \mu_0 \mathbf{1}$, ie, all assets have identical returns. In the second scenario, we allowed the assets to have different returns. Also, within each of these five test cases, we considered two test sets. In the first test set (set 1) the subsets of assets form a partition, whereas in the second test set (set 2) we considered overlapping subsets of assets that did not form a partition. The subsets $\mathcal{M}_k, k = 1, \dots, s$, for each test set were chosen randomly by generating an $s \times d$ -matrix M of zeros and ones. A nonzero entry, say $M_{(i,j)}$, then indicates that the j th asset belongs to the i th subset (\mathcal{M}_i). In the case of the first test set, nonzero entries that result in overlapping subsets of assets were simply set to zero so that the resulting subsets formed a partition.

In order to evaluate the AL-MCMC algorithm, we generated five antithetic pairs of random points. For each pair (x, x') of random points, we first sampled $x = (x_1, \dots, x_d)$ from $\mathbb{U}^d(0, 1)$ and set $x' = (1 - x_1, \dots, 1 - x_d)$. We used each of these ten points as the starting point x_0 to generate $T = 1000$ samples from the

¹⁴ It can readily be checked that x^* is indeed the optimal risk parity solution by solving Problem 2.3 directly.

¹⁵ All our experiments were performed using MATLAB on an Intel Core i5-680 (3.60 GHz), 64-bit operating system.

TABLE 1 Test case descriptions.

Test case	d	s	fmincon
1	7	3	3 000
2	30	5	5 000
3	50	5	7 000
4	100	10	20 000
5	200	10	60 000

d denotes number of assets. s denotes number of subsets. fmincon denotes the maximum number of function evaluations for the MATLAB fmincon parameter.

Markov chain. Therefore, we simulated a total of $N = 10$ Markov chains.¹⁶ The final point,¹⁷ ie, the T th point, from each chain is then used as the starting point of a nonlinear optimization solver. In our experiments we used the fmincon solver with the interior point method in MATLAB as our nonlinear optimization solver. The maximum number of function evaluations allowed for fmincon for each test case is specified in the final column of Table 1. We also used MATLAB and CVX (Grant and Boyd 2008, 2014) for solving the SDP relaxation of the GRB problem (2.8).

3.2.1 Identical returns: $\mu = \mu_0 \mathbf{1}$

In this scenario, we assume $\mu = \mu_0 \mathbf{1}$. In this case, the GRB problem reduces to the minimum risk problem subject to the risk budgeting constraints.

We solved the GRB problem using four different approaches. First, we solved the SDP relaxation to obtain the lower bound. Next, we used the possibly infeasible solution of the SDP relaxation as the initial point for fmincon. We refer to this as the SDP-fmincon approach. Next, we solved the problem using the AL-MCMC-fmincon approach, ie, we simulated ten Markov chains starting from ten random initial points generated using the antithetic random variate method, and used the $T = 1000$ th iterate of each chain as the initial point for a call to fmincon. In order to benchmark the contribution of the MCMC algorithm, we solved the GRB problem using fmincon starting from $10 \times T = 10\,000$ random starting points distributed

¹⁶ The values of N and T were chosen by testing various combinations on a training data set for each test case. In practice, the number of MCMC iterations typically ranges from 1000 to 10 000 combined across all chains used; eg, $N = 10$ chains with a length of $T = 1000$ each corresponds to $N \times T = 10\,000$ total iterations. We refer the reader to Section 4 for a discussion on how to choose chain lengths, T , and the number of chains, N .

¹⁷ The rationale behind choosing the final point is that due to risk budgeting constraints, sampled points are most likely to be infeasible, and an infeasible sample point attaining the highest value of $F(\cdot)$ is not necessarily the best point in terms of its proximity to the optimum solution.

according to $\mathbb{U}^d(0, 1)$.¹⁸ In addition, we also considered the alternating linearization backtracking (ALM-BTKR) approach to solve risk parity problems. The ALM-BTKR approach was introduced in Bai *et al* (2016), where the risk parity problem was formulated as the quadratic least-squares problem:

$$\min_{x \in \mathcal{X}, \theta} \sum_{k=1}^s \left(\sum_{i \in \mathcal{M}_k} x_i (\Sigma x)_i - \theta \right)^2, \quad \text{subject to } a_i \leq x_i \leq b_i, \quad i = 1, \dots, d. \quad (3.1)$$

Note that risk parity is achieved when (3.1) has an optimal value of zero. This approach can easily be extended to the case where the risks of the various asset classes \mathcal{M}_k are not equal by scaling θ as follows:

$$\min_{x \in \mathcal{X}, \theta} \sum_{k=1}^s \left(\sum_{i \in \mathcal{M}_k} x_i (\Sigma x)_i - \beta_k \theta \right)^2, \quad \text{subject to } a_i \leq x_i \leq b_i, \quad i = 1, \dots, d. \quad (3.2)$$

We stress that (3.1) and (3.2) are only able to identify a portfolio that satisfies the risk-parity constraints; they do not seek a risk-parity portfolio with minimum risk.

In Table 2, we report the following metrics for each of the four solution methods: SDP-fmincon, AL-MCMC-fmincon, fmincon and ALM-BTKR.

- $\min \tilde{F}(x^*)$ denotes the objective value of the best feasible portfolio computed by the methods. For the SDP-fmincon, we report the risk of the portfolio computed by fmincon starting from the SDP solution, if feasible. For the AL-MCMC-fmincon method we report the best objective value among all feasible solutions resulting from the ten starting points, and for the fmincon approach we report the best objective value among all feasible portfolios resulting from the $10 \times T = 10\,000$ random starting points. For ALM-BTKR, we report

$$\tilde{F}(x_{\text{ALM-BTKR}}^*),$$

where $x_{\text{ALM-BTKR}}^*$ is the solution obtained by solving (3.2).

- The range of objective values of all the feasible solutions computed by the method. We do not report a solution range for ALM-BTKR and SDP-fmincon, since these methods yield at most one feasible solution.
- The SDP lower bound.
- The number of failures. A failure occurs when fmincon does not return a feasible solution for a feasibility tolerance of 10^{-6} .

¹⁸ As solving GRB problems with fmincon was extremely time consuming, 1000 random starting points were used instead of 10 000 for test case 5 with 200 assets.

TABLE 2 Numerical results for the case of $\mu = \mu_01$. [Table continues on next page.]

(a) Test case 1: seven assets and three subsets						
Set	Method	$\min \tilde{F}(x^*)$	Solution range	SDP lower bound	# of failures	t
1	SDP-fmincon	33.17	—	31.72	0	3.64
1	AL-MCMC-fmincon	33.09	[33.09, 34.16]	31.72	0	18.50
1	fmincon	33.09	[33.09, 229.82]	31.72	194	100.26
1	ALM-BTKR	111.00	—	31.72	0	16.84
2	SDP-fmincon	25.76	—	25.57	0	2.06
2	AL-MCMC-fmincon	25.78	[25.78, 31.91]	25.57	0	16.90
2	fmincon	25.77	[25.77, 98.65]	25.57	840	1764.15
2	ALM-BTKR	N/A	—	25.57	1	21.37
(b) Test case 2: thirty assets and five subsets						
Set	Method	$\min \tilde{F}(x^*)$	Solution range	SDP lower bound	# of failures	t
1	SDP-fmincon	38.59	—	38.06	0	3.44
1	AL-MCMC-fmincon	38.59	[38.59, 38.59]	38.06	0	2.54
1	fmincon	38.59	[38.59, 328.61]	38.06	968	5.48
1	ALM-BTKR	100.61	—	38.06	0	49.39
2	SDP-fmincon	40.12	—	39.55	0	1.26
2	AL-MCMC-fmincon	40.12	[40.12, 40.12]	39.55	0	0.81
2	fmincon	40.12	[40.12, 40.12]	39.55	0	0.50
2	ALM-BTKR	N/A	—	39.55	1	63.71
(c) Test case 3: fifty assets and five subsets						
Set	Method	$\min \tilde{F}(x^*)$	Solution range	SDP lower bound	# of failures	t
1	SDP-fmincon	38.24	—	36.97	0	3.65
1	AL-MCMC-fmincon	38.24	[38.24, 38.24]	36.97	0	2.68
1	fmincon	38.25	[38.25, 328.33]	36.97	2339	4.17
1	ALM-BTKR	79.89	—	36.97	0	106.72
2	SDP-fmincon	64.33	—	51.77	0	2.71
2	AL-MCMC-fmincon	64.33	[64.33, 64.33]	51.77	0	1.90
2	fmincon	64.33	[64.33, 64.33]	51.77	0	1.41
2	ALM-BTKR	N/A	—	51.77	1	118.22

TABLE 2 Continued.

(d) Test case 4: one hundred assets and ten subsets						
Set	Method	$\min \tilde{F}(x^*)$	Solution range	SDP lower bound	# of failures	t
1	SDP-fmincon	55.33	—	53.76	0	14.84
1	AL-MCMC-fmincon	55.33	[55.33, 103.99]	53.76	0	7.98
1	fmincon	55.33	[55.33, 314.89]	53.76	3213	26.88
1	ALM-BTKR	110.39	—	53.76	0	354.29
2	SDP-fmincon	51.71	—	48.28	0	14.02
2	AL-MCMC-fmincon	51.71	[51.71, 51.71]	48.28	0	7.51
2	fmincon	51.71	[51.71, 51.71]	48.28	0	6.22
2	ALM-BTKR	N/A	—	48.28	1	423.02

(e) Test case 5: two hundred assets and ten subsets						
Set	Method	$\min \tilde{F}(x^*)$	Solution range	SDP lower bound	# of failures	t
1	SDP-fmincon	54.48	—	49.85	0	127.43
1	AL-MCMC-fmincon	54.48	[54.48, 54.68]	49.85	5	67.89
1	fmincon	54.48	[54.48, 393.48]	49.85	642	613.94
1	ALM-BTKR	N/A	—	49.85	1	2607.56
2	SDP-fmincon	56.02	—	49.85	0	127.69
2	AL-MCMC-fmincon	55.02	[55.02, 55.23]	49.85	0	54.87
2	fmincon	55.02	[55.02, 110.96]	49.85	58	114.84
2	ALM-BTKR	N/A	—	49.85	1	2378.06

Results for four methods: SDP relaxation, SDP-fmincon, AL-MCMC-fmincon, fmincon and ALM-BTKR for the case where $\mu = \mu_0 \mathbf{1}$. The first and second columns contain the test set number and the name of the algorithm, respectively. The third column reports the best solution obtained. For the ALM-BTKR method, this column reports $\tilde{F}(x_{\text{ALM-BTKR}}^*)$, where $x_{\text{ALM-BTKR}}^*$ is the solution obtained by solving (3.2) using the ALM-BTKR method. The fourth column reports the range of the obtained solutions. The fifth column reports the lower bound on the objective function $\tilde{F}(x)$ obtained by the SDP relaxation. The sixth column reports the number of failures. The last column reports the execution time (in seconds) required to obtain the best solution.

- t (in seconds). We report the amount of time taken to obtain $\min \tilde{F}(x^*)$ for the first time over the ten trials for the AL-MCMC-fmincon algorithm and over the 10 000 trials for the fmincon algorithm (see footnote 18). For ALM-BTKR and SDP-fmincon, t represents the total execution time, as each only yields a single solution.

The main issue with reporting the execution time t in the manner above is that, in order to determine the $\min \tilde{F}$, we first need to compute all ten solutions for the AL-MCMC-fmincon algorithm and 10 000 solutions for the fmincon algorithm. However,

given that AL-MCMC-fmincon uses only ten random points and fmincon uses 10 000 random points, comparing the total execution time of each method is fraught with difficulties. The advantage of reporting t in the above way is that it allows a fairer comparison of AL-MCMC-fmincon with fmincon. Note that all solutions are reported with a precision of four decimal places.

From the results reported in Table 2, the portfolio $x_{\text{ALM-BTKR}}^*$ is not the minimum variance portfolio. In most cases, $\tilde{F}(x_{\text{ALM-BTKR}}^*)$ is at least 50% larger than the solutions obtained by the other two methods. Also, due to the backtracking component of ALM-BTKR, the method may become prohibitively slow for higher-dimensional problems. For test case 1 with seven assets the ALM-BTKR method takes less than 25 seconds, but for test case 4 with 100 assets the execution time is more than 350 seconds. In fact, for test case 5 with 200 assets, ALM-BTKR fails to find a feasible solution even after 2350 seconds.

When all assets are assumed to have identical expected returns, the SDP relaxation appears to provide a fairly effective lower bound, against which we can compare solutions obtained from other methods. For example, in set 2 of test case 1, the difference between the SDP relaxation and the AL-MCMC-fmincon solution is just 0.21, and therefore we know that the AL-MCMC solution is close to the global optimum. Since the GRB problem is nonconvex, having an effective lower bound on its objective function is very informative. Moreover, the SDP-fmincon method exhibits comparable performance to the AL-MCMC-fmincon method in most cases, except for set 1 of test case 1, in which the optimal solution of the SDP-fmincon method is 0.08 higher than that of the AL-MCMC-fmincon method.

The AL-MCMC-fmincon method generally has better performance than fmincon. The execution time of fmincon is inconsistent. For example, the execution time of fmincon for test case 1 with seven assets ranges anywhere from 100 seconds to 1764 seconds to obtain the best solution. For set 1 of test cases 3 and 4, fmincon failed over 2000 and 3000 times, respectively. Similarly, for set 1 test case 5, it failed over 64% (642/1000) of the time. In contrast, the AL-MCMC-fmincon method was able to find an optimal solution within twenty seconds for test cases 1–4, and in most cases this took less than ten seconds. For test case 5, we observe a significant increase in execution time across all four methods. However, AL-MCMC-fmincon's increased execution time of about a minute is still superior to the other methods. In addition, for test cases 1–4, the AL-MCMC-fmincon method never failed, suggesting that the AL-MCMC approach is able to generate a good starting point for fmincon. The magnitude of constraint violations that caused its failures in set 1 test case 5 was, moreover, less than 10^{-4} . The range of values of the solutions computed by fmincon is also much wider than the range for AL-MCMC-fmincon. The test cases in which subsets form a partition, ie, set 1, clearly highlight the fact that fmincon significantly underperforms AL-MCMC-fmincon. For instance, in set 1 test case 4, the fmincon solution range is

TABLE 3 Numerical results for the SDP relaxation.

Set	Test case	d	s	Upper bound	Max. constraint violation	t
1	1	7	3	103.82	0.76	0.38
2	1	7	3	33.68	0.39	0.38
1	2	30	5	-22.73	0.24	0.58
2	2	30	5	28.38	0.63	0.98
1	3	50	5	5.41	0.52	0.92
2	3	50	5	-3.51	0.39	0.87
1	4	100	10	-15.62	0.10	6.29
2	4	100	10	3.25	0.23	6.66
1	5	200	10	24.05	0.27	87.38
2	5	200	10	24.05	0.35	70.21

The table presents the upper bound on the objective function, $F(x) = \mu'x - \mathcal{R}(x)$, of the GRB problem obtained via the SDP relaxation. The first and second columns contain the test set and case number, respectively. The third and fourth columns contain the number of assets (d) and the number of subsets (s) considered in each test case, respectively. The sixth column contains the maximum constraint violation of the optimal SDP solution, ie, (3.3), and the final column reports the execution time (in seconds) of CVX for solving the SDP relaxation.

[55.33, 314.89], whereas the AL-MCMC-fmincon solutions range is [55.33, 103.99]. This is actually the widest solution range we see for AL-MCMC-fmincon. In other test cases, solution ranges are very narrow for the AL-MCMC-fmincon algorithm.

We also compared the performance of the AL-MCMC algorithm against AL-MCMC but with the Lagrange multipliers $u_t \equiv 0$ for all t , ie, a pure penalty method, and AL-MCMC with the penalty parameter $c_t \equiv 0$ for all t , ie, a pure dual method. We found that the AL-MCMC-fmincon algorithm performed better overall than both of these alternatives. These results further demonstrate the merit of integrating the augmented Lagrangian method with the MCMC algorithm. We report the results for the AL-MCMC algorithm with $u_t \equiv 0$ and $c_t \equiv 0$ in Appendix B online.

3.2.2 General expected returns

The next set of results are for the case where expected returns are not identical across assets. Table 3 presents upper bounds for the GRB problem that were obtained using the SDP relaxation. We also report the maximum constraint violation of the optimal SDP solution, x_{SDP}^* :

$$\max_{k=1,\dots,s} \left| \sum_{i \in \mathcal{M}_k} \frac{\text{RC}_i(x_{\text{SDP}}^*)}{\mathcal{R}(x_{\text{SDP}}^*)} - \beta_k \right|. \quad (3.3)$$

Unlike in the previous section, we see that the maximum constraint violation of the solution of the SDP relaxation is quite large in certain cases. For example, it is 0.76

TABLE 4 Numerical results for the case of $\mu \neq \mu_0 \mathbf{1}$. [Table continues on next page.]

(a) Test case 1: seven assets and three subsets						
Set	Method	$\max \tilde{F}(x^*)$	Solution range	SDP upper bound	# of failures	t
1	SDP-fmincon	N/A	—	103.82	1	2.03
1	AL-MCMC-fmincon	63.89	[49.42, 63.89]	103.82	0	4.48
1	fmincon	63.88	[-235.38, 63.88]	103.82	990	6398.85
2	SDP-fmincon	30.31	—	33.68	0	0.88
2	AL-MCMC-fmincon	30.31	[30.31, 30.31]	33.68	0	0.58
2	fmincon	30.31	[-83.04, 30.31]	33.68	568	2.17

(b) Test case 2: thirty assets and five subsets						
Set	Method	$\max \tilde{F}(x^*)$	Solution range	SDP upper bound	# of failures	t
1	SDP-fmincon	-25.33	—	-22.73	0	1.91
1	AL-MCMC-fmincon	-25.33	[-25.33, -25.33]	-22.73	0	2.37
1	fmincon	-25.33	[-309.20, -25.33]	-22.73	640	15.13
2	SDP-fmincon	14.89	[14.89, 14.89]	28.37	0	1.50
2	AL-MCMC-fmincon	14.89	[14.89, 14.89]	28.37	0	0.68
2	fmincon	14.89	[-14.05, 14.89]	28.37	0	0.42

(c) Test case 3: fifty assets and five subsets						
Set	Method	$\max \tilde{F}(x^*)$	Solution range	SDP upper bound	# of failures	t
1	SDP-fmincon	N/A	—	5.41	1	N/A
1	AL-MCMC-fmincon	-0.30	[-0.30, -0.30]	5.41	0	2.52
1	fmincon	-0.30	[-345.38, -0.30]	5.41	2253	2.30
2	SDP-fmincon	-20.72	[-20.72, -20.72]	28.38	0	2.42
2	AL-MCMC-fmincon	-20.72	[-20.72, -20.72]	28.38	0	2.00
2	fmincon	-20.72	[-20.72, -20.72]	28.38	0	1.35

for set 1 test case 1, 0.63 for set 2 test case 2 and 0.52 for set 1 test case 3.¹⁹ In these cases, the SDP upper bound is likely to be slack since the SDP solutions are far from being feasible.

¹⁹ Recall that the β_k are all positive and sum to 1, so a violation of 0.76 is indeed quite large.

TABLE 4 Continued.

(d) Test case 4: one hundred assets and ten subsets						
Set	Method	$\max \tilde{F}(x^*)$	Solution range	SDP upper bound	# of failures	t
1	SDP-fmincon	-17.28	—	-15.62	0	13.97
1	AL-MCMC-fmincon	-17.27	[-17.28, -17.27]	-15.62	0	7.43
1	fmincon	-17.27	[-302.76, -17.27]	-15.62	1878	8.43
2	SDP-fmincon	-0.46	[-0.46, -0.46]	28.38	0	13.26
2	AL-MCMC-fmincon	-0.46	[-0.46, -0.46]	28.38	0	7.61
2	fmincon	-0.46	[-43.29, -0.46]	28.38	0	6.87

(e) Test case 5: two hundred assets and ten subsets						
Set	Method	$\max \tilde{F}(x^*)$	Solution range	SDP upper bound	# of failures	t
1	SDP-fmincon	8.41	—	24.05	0	135.16
1	AL-MCMC-fmincon	8.41	[-172.35, 8.41]	24.05	4	47.55
1	fmincon	8.42	[-379.73, 8.42]	24.05	535	735.36
2	SDP-fmincon	7.50	—	24.05	0	106.25
2	AL-MCMC-fmincon	7.52	[7.51, 7.52]	24.05	0	40.66
2	fmincon	7.52	[-98.34, 7.52]	24.05	6	296.05

Numerical results for the three methods: SDP-fmincon method, AL-MCMC-fmincon method and fmincon, when $\mu \neq \mu_0 \mathbf{1}$. The first and second columns contain the test set number and the name of the algorithm, respectively. The third column reports the best solution, ie, $\max F(x^*)$, and the fourth column reports the range of the obtained solutions. The fifth column reports the upper bound on the objective function $F(x)$ obtained by the SDP relaxation. The sixth column reports the number of failures. The final column reports the execution time (in seconds).

In Table 4, we report the following metrics for SDP-fmincon, AL-MCMC-fmincon and fmincon starting from 10 000 random points.

- $\max F(x^*)$ denotes the objective value of the best feasible solution computed by the three solution methods.
- The range of values for all feasible solutions. Note that we do not report a solution range for SDP-fmincon since this method produces only one solution.
- The SDP upper bound.
- A failure occurs when fmincon does not return a feasible solution at a feasibility tolerance of 10^{-6} .
- t (in seconds). We report the amount of time taken to obtain $\max F(x^*)$ for the first time in the ten trials for the AL-MCMC-fmincon algorithm and the 10 000 trials (see footnote 18) for the fmincon algorithm.

All solutions are reported with a precision of four decimal places. It is clear that the overall performance of the AL-MCMC-fmincon method is superior to the other two methods. In comparison with the SDP-fmincon approach, we note that the AL-MCMC-fmincon method was able to find an optimal solution to all test cases. The SDP-fmincon method failed to find even a feasible solution for set 1 test cases 1 and 3. It is interesting to note that these failures occurred when the maximum constraint violations of the SDP relaxation solutions were noticeably large (see Table 3). This suggests that an upper bound obtained from the SDP relaxation may turn out to be slack when the optimal SDP solution violates the risk budgeting constraints by a large amount. In the previous section, we saw that the SDP relaxation provides a relatively tight bound when all assets have identical expected returns.

In comparison with fmincon, AL-MCMC-fmincon exhibits a more stable and consistent performance. It is apparent from Table 4 that the solution ranges given by fmincon are very wide in general. For example, the objective values of the solutions obtained from fmincon for set 1 of test case 1 range from -235.38 to 63.89 , and it took 6398.85 seconds to discover the best of these solutions, despite test case 1 being a low-dimensional problem with only seven assets. This suggests that when fmincon uses an unfavorable starting point the solution it obtains can be very far from a good local optimum. The results of set 1 test case 3 further demonstrate this. Of the 10 000 random starting points, fmincon failed to find a feasible solution 2253 times. In many cases, the stand-alone fmincon has a relatively large number of failures compared with AL-MCMC-fmincon, and hence we need to try a very large number of random starting points. In contrast, the AL-MCMC-fmincon method yields solutions with much narrower ranges. Except for set 1 of test cases 1 and 5, all the solutions obtained are very close to the best solution. Moreover, except for set 1 test case 5, the AL-MCMC-fmincon method never fails to produce a feasible point. This means that the starting points generated by the AL-MCMC algorithm are much more favorable than random starting points. Note that even the failures in set 1 test case 5 were caused by the constraint violations, which were less than 10^{-4} . Also note that in most cases AL-MCMC-fmincon finds an optimal solution in less than ten seconds. For test case 5, AL-MCMC-fmincon solves the GRB problem within fifty seconds, which is quite exceptional compared with the other two methods. Based on these observations, we can conclude that AL-MCMC-fmincon appears to be a much more reliable tool for solving the general GRB problem with nonidentical expected asset returns.

3.3 Empirical results for the GRB problem

As noted in Section 2, when evaluating the risk of a large-scale equity portfolio it is often advisable to group stocks according to attributes, such as market sector. Accordingly, risk-based sector weighting is a good example of a risk budgeting strategy that

TABLE 5 Empirical testing results.

Method	# of successes	Mean % deviation from v^*	Average failure rate (%)	Average time (s)
SDP-fmincon	5	—	77.50	109.94
AL-MCMC-fmincon	120	77.16	21.33	137.19
fmincon	96	97.31	49.92	192.21

Empirical testing results for the three methods: SDP-fmincon, AL-MCMC-fmincon and fmincon, when $\mu = \mu_0 \mathbf{1}$. The first column contains the name of the algorithm. The second column reports the number of successes, and the third column reports the mean percentage deviation of a feasible solution from the optimal objective value v^* . The fourth column reports the average failure rate (in percent), and the final column reports the average execution time (in seconds).

can be built using GRB models in practice. We now present some empirical results for such an application.

We took our security universe to be the 200 largest (by market capitalization) stocks in the Standard & Poor's 500. We assigned these stocks to ten sectors (consumer discretionary; consumer staples; energy; financials; healthcare; industrials; information technology; materials; telecommunications services; utilities) according to the Global Industry Classification Standard (GICS) and estimated the monthly covariance matrix over the past ten years (520 weeks) from August 5, 2005 to July 17, 2015.²⁰ Our goal in each month was to compute a minimum variance portfolio that satisfies the predefined risk budgeting constraints at a market sector level.²¹ We therefore have a total of 120 empirical test cases. We note that these test cases are equivalent to set 1 test case 5 presented in Section 3.2.1. We also note that, due to the large number of test cases, we use ten instead of 10 000 randomly generated starting points for fmincon, and keep all other testing parameters the same as those used in our earlier numerical experiments.

In Table 5, we report the following metrics for each of the three solution methods (SDP-fmincon, AL-MCMC-fmincon and fmincon).²²

The number of successes. The i th trial is deemed a success for a given method if the method achieves the minimum objective value v_i^* on that trial. For SDP-fmincon the objective value is defined as the risk of the portfolio computed by fmincon starting from the SDP solution, provided the SDP solution is feasible. For

²⁰ To estimate the covariance matrix in any given month, we used the previous two years of weekly returns and fitted a one-factor model based on the first principal component of the return data. For further details on the covariance matrix estimation using factor models, see Ruppert (2010).

²¹ Risk budgets $\beta = (\beta_1, \dots, \beta_{10})$ are generated as $\mathbb{U}^{10}(0, 1)$ and then normalized so that $\sum_{k=1}^{10} \beta_k = 1$.

²² ALM-BTKR is not included in our empirical testing as its execution time for the 200-asset case exceeded 2350 seconds, and despite such an execution time it failed to find a feasible portfolio.

AL-MCMC-fmincon and fmincon the objective value is defined as the lowest risk of all feasible solutions resulting from the ten starting points.

The mean percentage deviation (MPD) from the optimal objective value v^* . The MPD is defined as

$$\text{MPD} = \frac{1}{120} \sum_{i=1}^{120} \frac{1}{|P_i|} \sum_{j \in P_i} \frac{\tilde{F}(x_{i,j}) - v_i^*}{v_i^*},$$

where P_i is the set of feasible portfolios, $x_{i,j}$, computed by the method under consideration from the ten starting points and v_i^* is the best objective value on the i th test case. We do not report this metric for SDP-fmincon since this method yields at most one feasible solution.

The average failure rate (in percent). A failure occurs when fmincon does not return a feasible solution for a feasibility tolerance of 10^{-6} . For AL-MCMC-fmincon and fmincon, we report the average percentage of times feasible solutions were not found in the ten trials across 120 test cases. For SDP-fmincon, we report the average percentage of times feasible solutions were not found across 120 test cases.

The average time (in seconds). For AL-MCMC-fmincon and fmincon, we report the average amount of time taken to obtain the optimal objective value for the first time in the ten trials across 120 test cases. For SDP-fmincon, t represents the average execution time across 120 test cases, as each only yields a single solution.

AL-MCMC-fmincon is successful in all test cases, ie, the risk of the solution computed by AL-MCMC-fmincon is the lowest for each test case. Moreover, for most test cases, AL-MCMC-fmincon found an optimal solution in the shortest time. Although the reported average execution time for SDP-fmincon is less than that of AL-MCMC-fmincon, it does not account for the fact that SDP-fmincon failed to find a feasible solution ninety-three times out of 120 test cases. The failure rate of AL-MCMC-fmincon (21.33%) is significantly lower than that of the other two methods (77.50% and 49.92% for SDP-fmincon and fmincon, respectively). In terms of solution quality, the MPD from v^* for AL-MCMC-fmincon is approximately 20% lower than that for fmincon. This shows that AL-MCMC-fmincon is better suited for solving the GRB than fmincon.

The empirical results presented here are consistent with the numerical results presented in Section 3. The AL-MCMC-fmincon method is a more robust and dependable approach for solving the GRB problems.

4 DISCUSSION

It is well known that nonlinear optimization methods perform poorly, and can even fail, if the starting point is poorly chosen. When solving a multimodal problem, in particular, these methods could get easily trapped by a (poor) local optimum due to an unfavorable starting point. The AL-MCMC approach proposed in this study combines the augmented Lagrangian, MCMC and simulated annealing to avoid this phenomenon. We expect that the AL-MCMC method can be adapted to solve other multimodal problems.

The AL-MCMC method has several parameters that need to be specified, including, but not limited to, the chain length, the number of chains, the variance σ_t^q of the proposal distribution and the annealing schedule. As with any numerical method, the performance of AL-MCMC depends on the choice of parameters. For example, the selection of σ_t^q can greatly affect the performance of the sampler. The most widely used approach to choosing σ_t^q is the trial-and-error approach. In this approach, initial values of the tuning parameters are chosen and then modified based on the examination of the mixing properties of the chains.²³ For instance, although the overall acceptance rate in our numerical experiments was around 85%, we initially explored the parameter space more freely by allowing (relatively) large jumps and gradually decreasing σ_t^q as we made progress toward the feasible region, ie, we adjusted the acceptance probability by allowing the tuning parameter to vary with t . According to Bounds (1987), “choosing an annealing schedule for practical purposes is still something of a black art” despite the simple principle underlying the choice of a suitable annealing schedule: the initial temperature should be low enough to “freeze” the system and should be increased toward its boiling point as the search progresses. Nevertheless, for some parameters we can certainly make a more informed decision. For multimodal problems, for example, using multiple (relatively) short chains would yield a better performance than using a single long chain.²⁴

5 CONCLUSIONS

In this paper, we proposed a GRB approach to portfolio construction. Our approach provides investors with more flexibility than the existing risk-based asset allocation techniques, in that it allows investors to optimize a risk–return profile and to define

²³ For a discussion of this procedure in the context of MCMC, we refer the reader to Gelman *et al* (2004).

²⁴ Using multiple highly dispersed initial values to start several different chains is the most straightforward approach to solving a multimodal problem (Gelman and Rubin 1992), and has a computational advantage for parallel processing machines.

risk budgets for possibly overlapping subsets of assets. Minimum variance, risk parity and risk budgeting strategies are therefore special cases of GRB strategies.

Although we show that the GRB problem can be formulated as a convex optimization problem in an important special case, the general GRB problem is a nonconvex optimization problem. We introduce an SDP relaxation for bounding the optimal value of the GRB problem. When all assets have identical expected returns, our numerical results suggested that this SDP bound was quite tight, and could therefore be used to assess the quality of solutions produced by other approaches. Our main contribution in this paper is a simulation-based algorithm that combines augmented Lagrangian optimization ideas with MCMC methods. The goal of this algorithm is to compute a candidate solution in the neighborhood of the optimum, or a very good local optimal solution of the GRB problem. This candidate solution could then be used as the starting point for a standard nonlinear optimization solver. In several numerical experiments our AL-MCMC algorithm was indeed successful in finding very good starting points.

We also note that our AL-MCMC approach is a general solution approach for solving nonconvex optimization problems. The augmented Lagrangian algorithm is a very popular algorithm for computing local optimum solutions for nonconvex problems. Combining this algorithm with the MCMC method opens up the possibility of converging to the global optimal solution, or at least providing a good starting point for a nonlinear optimization routine. In addition, this approach can be implemented very easily and is computationally fast.

We expect it to be of particular use for nonconvex problems with small feasible regions, where computing a good starting point is challenging. We intend to apply this approach to such problems in future research.

DECLARATION OF INTEREST

The authors report no conflicts of interest. The authors alone are responsible for the content and writing of the paper.

REFERENCES

- Artzner, A., Delbaen, F., Eber, J.-M., and Heath, D. (1999). Coherent measures of risk. *Mathematical Finance* **9**(3), 203–228 (<http://doi.org/bzjp2p>).
- Bai, X., Scheinberg, K., and Tutuncu, R. (2016). Least-squares approach to risk parity in portfolio selection. *Quantitative Finance* **16**(3), 357–376 (<http://doi.org/bwz3>).
- Bertsekas, D. P. (1996). *Constrained Optimization and Lagrange Method*. Athena Scientific, Belmont, MA.
- Black, F., and Litterman, R. (1992). Global portfolio optimization. *Financial Analysts Journal* **48**, 28–43 (<http://doi.org/d79fhm>).

- Boudt, K., Carl, P., and Peterson, B. G. (2012). Asset allocation with conditional value-at-risk budgets. *The Journal of Risk* **15**(3), 39–68 (<http://doi.org/bwz4>).
- Bounds, D. G. (1987). New optimization methods from physics and biology. *Nature* **329**, 215–218 (<http://doi.org/dt7pvq>).
- Boyd, S., and Vandenberghe, L. (1997). Semidefinite programming relaxations of non-convex problems in control and combinatorial optimization. In *Communications, Computation, Control and Signal Processing: A Tribute to Thomas Kailath*, pp. 279–288. Kluwer.
- Bruder, B., and Roncalli, T. (2012). Managing risk exposures using the risk budgeting approach. Working Paper, Social Science Research Network (<http://doi.org/bwz5>).
- DeMiguel, V., Garlappi, L., and Uppal, R. (2009). Optimal versus naive diversification: how inefficient is the $1/N$ portfolio strategy? *Review of Financial Studies* **22**, 1915–1953 (<http://doi.org/dzkc89>).
- Gelman, A., and Rubin, B. D. (1992). Inferences from iterative simulation using multiple sequences. *Statistical Science* **7**, 457–472 (<http://bit.ly/2ic0a4z>).
- Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2004). *Bayesian Data Analysis*. CRC Press, Boca Raton, FL.
- Grant, M., and Boyd, S. (2008). Graph implementations for nonsmooth convex programs. In *Recent Advances in Learning and Control*, Blondel, V., Boyd, S., and Kimura, H. (eds), pp. 95–110. Lecture Notes in Control and Information Sciences. Springer (<http://stanford.io/2ieyqzj>).
- Grant, M., and Boyd, S. (2014). CVX: MATLAB software for disciplined convex programming. Version 2.1, March (<http://bit.ly/2iY1mdS>).
- Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science* **220**, 671–680 (<http://doi.org/cn7jh2>).
- Maillard, S., Roncalli, T., and Teiletche, J. (2010). The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management* **36**(4), 60–70 (<http://doi.org/dsq7g6>).
- McNeil, A. J., Frey, R., and Embrechts, P. (2005). *Quantitative Risk Management: Concepts, Techniques, and Tools*. Princeton University Press.
- Nourani, Y., and Andresen, B. (1998). A comparison of simulated annealing cooling strategies. *Journal of Physics A: Mathematical and General* **31**, 8373–8385 (<http://doi.org/fj4bmj>).
- Qian, E. (2006). On the financial interpretation of risk contribution: risk budgets do add up. *Journal of Investment Management* **4**(4), 41–51 (<http://bit.ly/2iXW2XZ>).
- Robert, C. P., and Casella, G. (2004). *Controlling Monte Carlo Variance*. Springer.
- Roberts, G. O., Gelman, A., and Gilks, W. R. (1994). Weak convergence and optimal scaling of random walk metropolis algorithms. Working Paper, Statistical Laboratory, University of Cambridge.
- Ruppert, D. (2010). *Statistics and Data Analysis for Financial Engineering*. Springer.
- Spinu, F. (2013). An algorithm for computing risk parity weights. Working Paper, Social Science Research Network (<http://doi.org/bwz7>).
- Van Laarhoven, P. J., and Arts, E. H. L. (1987). *Simulated Annealing: Theory and Applications*. Reidel, Amsterdam (<http://doi.org/bwz8>).

Zhu, S. S., Li, D., and Sun, X. L. (2010). Portfolio selection with marginal risk control. *Journal of Computational Finance* **14**(1), 3–28 (<http://doi.org/bwz9>).

Zhu, S. S., Cui, X. T., Sun, X. L., and Li, D. (2011). Factor-risk-constrained mean–variance portfolio selection: formulation and global optimization solution approach. *The Journal of Risk* **14**(2), 51–89 (<http://doi.org/bw2b>).